# Problem J. Independent Set

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Time limit: | 1 second |
| Memory limit: | 1024 mebibytes |

*This is an interactive problem. You have to use the `flush` operation right after printing each line. For example, you can use the function `fflush(stdout)` for C or C++, `System.out.flush()` for Java, `flush(output)` for Pascal, and `sys.stdout.flush()` for Python.*

An independent set in a graph is a set of vertices such that, for every two vertices in the set, there is no edge connecting them.

There is an algorithm to construct an independent set in the graph.

The algorithm receives a sequence of vertices. Suppose the sequence is $a_1, a_2, \ldots, a_\ell$. Initially, there is an empty set $S$ and a sequence $cnt_1, cnt_2, \ldots, cnt_\ell$ that satisfies $cnt_1 = cnt_2 = \ldots = cnt_\ell = 0$.

Then for $i$ from 1 to $\ell$:

- Look at every edge in the graph, and increment $cnt_i = cnt_i + 1$ every time when the edge connects $a_i$ and a vertex in $S$.
- If $cnt_i = 0$ after searching, insert $a_i$ into $S$.

Obviously, $S$ will be an independent set after the algorithm.

Now, you only know the number of vertices $n$ in the graph, and you want to find all its edges. There is an interactor to help you. The interactor receives a sequence of vertices, runs the algorithm above, and returns the sequence $cnt_1, cnt_2, \ldots, cnt_\ell$.

You want to achieve your goal with the total length of the sequences that you feed no more than 176 000.

Note that there may be multiple edges and self-loops in the graph.

## Input

The first line contains a single integer $n$, the number of vertices in the graph.

Suppose $m$ is the number of edges in the graph. It is guaranteed that $1 \le n \le 4000$ and $0 \le m \le 10\,000$.

## Interaction Protocol

To feed a sequence $a_1, a_2, \ldots, a_k$ to the interactor, print a single line formatted as "`?` $k$ $a_1$ $a_2$ $\ldots$ $a_k$" ($k \ge 1$). Then you should read a single line with the answer, which is formatted as "$cnt_1$ $cnt_2$ $\ldots$ $cnt_k$".

If you have found all the edges in the graph, print a single line formatted as "`!` $m$ $x_1$ $y_1$ $x_2$ $y_2$ $\ldots$ $x_m$ $y_m$". The edges $(x_1, y_1)$, $(x_2, y_2)$, $\ldots$, $(x_m, y_m)$ denote your answer. You can print the edges in any order.

After printing each line, your program must perform the `flush` operation.

## Example

| *standard input* | *standard output* |
|---|---|
| `4`<br><br>`0 2 1 0 1 0`<br><br>`0 1 1 0 0` | <br>`? 6 1 2 3 1 3 4`<br><br>`? 5 4 4 4 2 3`<br><br>`! 4 1 2 1 2 1 3 4 4` |